AMENDMENTS TO THE CLAIMS

Claims 8, 11, 20-21, 23 and 31 have been amended. Claims 40-42 have been added. No claims have been cancelled in this Reply.

1. – 7. Cancelled

8. (Currently amended) A method of creating an image utilizing a central processing unit (CPU) and a graphics processing unit (GPU)~~, said image represented by an image graph, said image graph comprising one or more GPU programs, inputs to those programs and outputs from those programs~~, the method comprising the steps of:

representing, in memory, an image by an image graph wherein the image graph comprises one or more GPU programs, inputs to the one or more GPU programs and outputs from the one or more GPU programs;

optimizing said image graph by running software on a CPU;

compiling said image graph by running software on said CPU; and

rendering said image graph by running said compiled image graph on a GPU, yielding a rendered image.

9. (Original) The method of claim 8, wherein the step of optimizing includes the step of using a cache look-up to see if said rendered image is already in cache.

10. (Original) The method of claim 8 wherein the step of optimizing includes the step of using a cache look-up to see if said image graph has already been optimized and is in a memory.

11. (Currently amended) The method of claim 8, wherein the step of optimizing includes the step of [[C]]calculating an intersection, said intersection representing an area where said rendered image is both defined by said image graph and part of a region requested by a process running on said CPU that has requested creation of said image.

12. (Previously presented) The method of claim 9, wherein the step of optimizing includes the step of calculating an intersection, said intersection representing an area where said rendered image is both defined by said image graph and part of a region requested by a process running on said CPU that has requested creation of said image.

13. (Previously presented) The method of claim 10, wherein the step of optimizing includes the step of calculating an intersection, said intersection representing an area where said rendered image is both defined by said image graph and part of a region requested by a process running on said CPU that has requested creation of said image.

14. (Original) The method of claim 11 further comprising the step of, using said calculated intersection to limit the number of pixels that require calculation during said rendering on said GPU.

15. (Original) The method of claim 12 further comprising the step of, using said calculated intersection to limit the number of pixels that require calculation during said rendering on said GPU.

16. (Original) The method of claim 13 further comprising the step of, using said calculated intersection to limit the number of pixels that require calculation during said rendering on said GPU.

17. (Original) The method of claim 11 further comprising the step of, using said calculated intersection to limit the amount of memory necessary for storing said rendered image.

18. (Original) The method of claim 12 further comprising the step of, using said calculated intersection to limit the amount of memory necessary for storing said rendered image.

19. (Original) The method of claim 13 further comprising the step of, using said calculated intersection to limit the amount of memory necessary for storing said rendered image.

20. (Currently amended) The method of claim 8 wherein said step of optimizing comprises the additional steps of

using a cache to determine if said rendered image is available in memory

using the CPU to perform [[ROI /DOD]]region of interest (ROI) and domain of definition (DOD) intersections with respect to one or more of said GPU programs;

using the CPU to determine if GPU programs may be combined; and

using the CPU to determine if said GPU is capable of running a program that has been created by combining two other GPU programs.

21. (Currently amended) The method of claim 8 wherein said step of optimizing comprises the additional steps of

using the CPU to perform [[ROI/DOD]]region of interest (ROI) and domain of definition (DOD) intersections with respect to one or more of said GPU programs;

using the CPU to determine if GPU programs may be combined; and

using the CPU to determine if said GPU is capable of running a program that has been created by combining two other programs.

22. (Previously presented) The method of claim 8 wherein said step of optimizing comprises the additional steps of

using the CPU to determine if GPU programs may be combined; and

using the CPU to determine if said GPU is capable of running a program that has been created by combining two other programs.

23. (Currently amended) A method for creating a rendered polygon, the method comprising the steps of:

under control of a CPU, receiving a request to render a polygon;

under control of said CPU, creating a representation of said rendered polygon comprising a root GPU program and its relationship with other GPU programs, their inputs and outputs;

under control of said CPU, starting with the root GPU program, calling the following groups of objects for each GPU program that must be run in order that the root GPU program may run to render said polygon;

one or more objects for analyzing whether two GPU programs may be combined, and performing a combination if said analysis is positive;

one or more objects for performing [[DOD/ROI]]region of interest (ROI) and domain of definition (DOD) optimization; and

one or more objects for creating a buffer and causing a GPU to render an image to that buffer by running a GPU program.

24. (Original) The method of claim 23 wherein said representation of said rendered polygon is a graph.

25. (Original) The method of claim 23 wherein said representation of said rendered polygon is a low-level graph.

26. (Original) The method of claim 23 wherein said representation of said rendered polygon is a high-level graph.

27. (Original) The method of claim 23 further wherein an application program under CPU control makes said request to render said polygon.

28. (Original) The method of claim 23 wherein said one or more objects for analyzing whether two GPU programs may be combined, and performing a combination if said analysis is positive, are recursively called until no more combinations are possible.

29. (Original) The method of claim 23 wherein DOD/ROI optimization comprises intersecting the ROI with the output DOD for a GPU program.

30. (Previously presented) The method of claim 23 wherein a cache is used in order to find the result of running a particular GPU program, without running the particular GPU program.

31. (Currently amended) A method for creating a rendered image, the method comprising the steps of:

under control of said CPU, creating a graph representing said rendered image wherein said graph comprises a representation of one or more GPU programs, inputs to the one or more GPU programs and outputs from the one or more GPU programs;

under control of said CPU, starting with a root node in said graph, calling the following groups of objects for each node that must be calculated in order that the root node may be calculated;

one or more objects for analyzing whether two GPU programs may be combined, and performing a combination if said analysis is positive;

one or more [[DOD/ROI]]domain of definition (DOD) and/or region of interest (ROI) objects for performing DOD/ROI optimization; and

one or more objects for creating a buffer and causing a GPU to render an image to that buffer by running a GPU program.

32. (Original) The method of claim 31 wherein said graph is a low-level graph.

33. (Original) The method of claim 31 wherein said graph is a high-level graph.

34. (Original) The method of claim 31 further comprising the step of receiving a request to render said image.

35. (Original) The method of claim 34 further comprising the step of an application program under CPU control making said request to render said image.

36. (Previously presented) The method of claim 31 wherein said one or more objects for analyzing whether two GPU programs may be combined, and performing a combination if said analysis is positive, are recursively called until no more combinations are possible.

37. (Original) The method of claim 31 wherein DOD/ROI optimization comprises intersecting the ROI with the output DOD for a GPU program.

38. (Previously presented) The method of claim 31 wherein a cache is used in order to find the result of computed particular node without computing the particular node.

39. (Previously presented) A computer-readable medium having computer executable instructions for performing the method recited in any one of claims 8, 23 or 31.

40. (New) A computer system configured for creating an image, the computer system comprising:

a central processing unit (CPU);

a graphics processing unit (GPU) communicatively coupled to the CPU; and

a memory communicatively coupled to the CPU and/or the GPU having, the memory storing computer executable instructions executable by the CPU and/or the GPU to configure the CPU and GPU to perform the method recited in claim 8.

41. (New) A computer system configured for creating an image, the computer system comprising:

a central processing unit (CPU);

a graphics processing unit (GPU) communicatively coupled to the CPU; and

a memory communicatively coupled to the CPU and/or the GPU having, the memory storing computer executable instructions executable by the CPU and/or the GPU to configure the CPU and GPU to perform the method recited in claim 23.

42. (New) A computer system configured for creating an image, the computer system comprising:

a central processing unit (CPU);

a graphics processing unit (GPU) communicatively coupled to the CPU; and

a memory communicatively coupled to the CPU and/or the GPU having, the memory storing computer executable instructions executable by the CPU and/or the GPU to configure the CPU and GPU to perform the method recited in claim 31.